PCT AS PUBLISHED

METHOD FOR SOFTWARE EMULATION OF HARD DISKS OF A DATA PROCESSING PLATFORM AT THE LEVEL OF THE OPERATING SYSTEM WITH PARAMETRIZABLE ON-THE-FLY MANAGEMENT OF WRITE AND READ REQUESTS

The present invention concerns a method for software emulation of hard disks of a data processing platform at the operating system level with parametrizable on-the-fly management of write and read requests.

In particular, it permits the complete emulation of one or more emulated hard disks, perceived as real hard disks by the operating system of a data-processing platform such as a personal computer (PC); the data contained in an emulated hard disk may be stored on any type of support such as a server on a network, a CD-R, a DVD-ROM, a flash memory or an external hard disk.

The invention also permits the complete emulation of a hard disk system, i.e. a hard disk permitting the booting up of the computer followed by the complete loading of the operating system whose components are stored on said emulated hard disk.

Generally speaking, it is known that in the microcomputers presently in use the operating system, the application programs as well as the data belonging

to the users are stored on a hard disk accessible through the central unit of the computer.

The hard disks appear as stacks of several plates whose different faces are accessible by head combs; sectors are recorded on each track.

For examples, for the IBM PC compatible microcomputers, the BIOS (basic input/output system) reads or writes the information coming from or going to the hard disk using a program loop. The application programs may use the BIOS for reading or writing on the hard disk either directly at the sector level via "int 13h" software interrupt or more frequently through the file system via "int21h" software interrupt.

The first sector of the hard disk contains a special program loaded in the memory and launched by the BIOS upon the startup of the computer. This program, called "MBR" (master boot record) controls the possible partitioning of the disk into several subspaces called "partitions." These "partitions" may be allotted to different operating systems that the user may select when starting up the computer. Once the choice is made, a program (called "boot") contained in the first sectors of each partition is loaded in turn and launched. Its task is to load the operating system stored on the corresponding partition of the hard disk and launch it. The following sectors serve to manage the space reserved in the files.

Therefore, when started up, the microcomputer proceeds to load the operating system contained in the hard disk, then the application software as well as the user data into its random access memory (RAM).

It is proven that during these startup operations the slightest anomaly often causes a destabilization resulting in a freeze-up of the micromethodor.

In addition, for the purpose of security in many cases the access to the operating system and/or certain files contained in the hard disk is only possible by certain users identified after the entering and recognition of a password (identifier). It has been found in practice that in a system operated as a network

by several users, this type of security increasingly causes problems due to the complexity of its management.

The invention, therefore, more particularly, has the objective of eliminating these shortcomings by devising a method which totally emulates the software of hard disks at the level of the data blocks, also called sectors, or at the level of the file system, therefore permitting the use of file systems accepted by the operating system in emulated hard disks of any type.

According to the invention this method is particularly characterized by the fact that it consists of creating in a first step a representation of a real hard disk in which the orders of loading and execution of certain components of the operating system of a data processing platform may be adapted, then in a second step loading onto said data processing platform one or more peripheral drivers, among which at least one of the peripheral drivers permits real dialogue with a data storage support containing the data of the emulated hard disk, then simulating in a third step the behavior of a real hard disk for the operating system.

As regards the term "support" used here, here and in the remaining description this refers to any system permitting the volatile or non-volatile saving of processing data; this term equally denotes a hard disk, a CD-ROM, a magnetic tape or a data service accessible through a computer network.

In addition, the invention provides for a specific processing of the writing operations performed on the emulated hard disk. For example, if the data storage support of said emulated hard disk is not an optionally rewritable medium, e.g. a CD-ROM, the recorded data may be stored in a read-write memory or on a support accepting write operations on the fly such as a real local hard disk. The specific processing of the written data permits the proposal of "volatile" emulated hard disks, i.e. not retaining changes made in the emulated hard disk after rebooting of the emulation, of standard emulations, where the emulated hard disk behaves exactly like a normal hard disk. Other types of management of written data are provided and permit other uses of the invention.

The invention also envisions, if the data storage support containing the data of the emulated hard disk permits, e.g., if the support is a file stored on a server of a network, that several computers may access the emulated hard disk simultaneously through said emulation. In this case, a method of distribution called "multicast" could be used so as to signficantly reduce the bandwidth used for the quasi-simultaneous readings requested by several "client" computers, which are carried out on the same parts of the emulated hard disk.

Thanks to these arrangements the invention permits the solution of the problems of destabilization of the startup of the data processing platform and the security problems mentioned above and greatly simplifies the administration of the data processing platforms.

In addition, in the case of a network system operated by several users, each station is no longer tied to applications loaded locally nor to regular users of the platform. In addition, the errors made by the users no longer have any effect on the overall functioning of the network system.

Thus, more precisely, the method according to the invention permits the dialogue between the emulated hard disk and the operating system «in the manner» of a real hard disk thanks to the intermediacy of one of the peripheral drivers mentioned above, by simulating the behavior of a real hard disk for the operating system in all of the phases of the startup process as well as during the phase traditionally reserved to the micro-software (BIOS for example) and during the phases handled by the operating system itself, in particular, by the components of the hard disk management of the operating system. This total emulation, including the possibility of startup, may require adjustments of the orders of loading and execution of certain of the operating system components.

Management of the data write requests that the operating system issues to the emulated hard disk is accomplished at the peripheral driver level and/or at the level of an optional server service of the hard disk on the network. This management permits the written data to be stored, depending on the parameter

settings of said peripheral drivers and/or of said hard disk server service on the network.

- either directly in the support containing the emulated hard disk
- or in the memory, random access or virtual, accessible to the operating system using the emulated hard disk,
- or in a volatile storage space accessible to the operating system using the emulated hard disk,
- or in a non-volatile storage space accessible to the operating system using the emulated hard disk,
- or in a volatile storage space accessible to the server service of emulated hard disks on a data processing network.
- or in a non-volatile storage space accessible to the emulated hard disk server service on a data processing network.

Management of the data read requests that the operating system issued to the emulated hard disk is accomplished at the peripheral driver level and/or at the level of an optional hard disk server service on the network in such a way that for a client station, the readings of previously written data are performed in the appropriate storage space, i.e.:

- either directly in the support containing the emulated hard disk,
- or in the random access or virtual memory accessible to the operating system using the emulated hard disk,
- or else in a volatile storage space accessible to the operating system using the emulated hard disk,
- or in a non-volatile storage space accessible to the operating system using the emulated hard disk,
- or in a volatile storage space accessible to the emulated hard disk server service on a data processing network,
- or in a non-volatile storage space accessible to the emulated hard disk server service on a data processing network.

In the case when the data of the emulated hard disk(s) are accessible to client stations via a data processing network, a specific program called "server software" is in charge on one of the stations of the data processing network, on the one hand, of the communications via the network with the client stations accessing the emulated hard disks, and on the other, of accessing the data support containing the data of the emulated hard disks.

If the emulated hard disk is to be started up, a low-level micro-software module is responsible for access to the data contained in the emulated hard disk. Said software module provides an interface of the micro-software type (BIOS, for example) in order to permit the good execution of the functions of loading the components of the operating system traditionally loaded by the microsoftware during bootup of the operating system from a real hard disk.

This micro-software could, in the case of PC compatible computers, use a 13 h interrupt manager, thus providing the operating system with the interface used by the latter.

The micro-software could, in the case of computers using bootup memory programs of the PXE type (PXE bootup PROM), use the functions made available by these PROMS for controlling communications via the data processing network independently of the network interface model employed.

The low level micro-software is loaded in the memory of the client station and executed by using the functions made available by a bootup PROM, such as a PXE starting chip, for example.

The low level micro-software may also be loaded in the memory of the client station and executed as a basic micro-software component (BIOS, for example) of the client station. In particular, said low-level micro-software performs the same functions as the access services on real hard disks usually provided by the standard basic micro-software. For example, in the case when the data contained in the emulated hard disk are stored on an optical disk such as a CD or a DVD, the low-level micro-software could be provided by the

manufacturer of the client station as part of the BIOS, providing the BIOS startup functions from an optical disk.

The low-level micro-software may finally be loaded in the memory of the client station from a third party data support supported as a startup peripheral by the client station, such as a diskette, a real hard disk or an optical disk, then executed by the client station.

It may be noted that if some of the data contained in the emulated hard disk are accessed by a client station via a data processing network, at least one peripheral driver, loaded and executed by the operating system of the client station, provides the functions of communication via the data processing network with the server software that is charged with furnishing the hard disk emulation services.

If the data support containing the data of the emulated hard disk(s) is a support that does not support writing in real time, e,g., an optical CD-ROM disk, or [if] the system of hard disk emulation according to the invention is parameterized not to accept data writing directly in the support containing the data of the emulated hard disk, e.g., an emulated hard disk used simultaneously by several client stations, the peripheral drivers, and /or a server software executed by a network data processing station providing the emulation of the hard disk at the client stations, may method the data write requests issued by the operating system to the emulated hard disk(s) in such a way that the written data are stored in a storage space different from the data support containing the data of the emulated hard disk(s).

For example, the data write requests issued by the client station operating system to the emulated hard disk(s) are processed in such a way that the written data are stored in the random access memory of the client station.

The data write requests issued by the operating system of the client station to the emulated hard disk(s) may either be processed in such a way that the written data are stored in the virtual memory of the client station, i.e. in a data file

accessible to the operating system of the client station and serving as additional memory space, or processed in such a way that the written data are stored in a data file accessible to the operating system of the client station.

The data write requests issued by the operating system to the emulated hard disk(s) are redirected at a given moment to one and only one storage space. The storage space to which the written data are redirected may be changed on the spur of the moment during an operating session of the operating system of the client station. The storage space used for storage of the written data may be volatile, i.e. dump the stored data upon each new operating session of the client station operating system, or nonvolatile so as to permit the written data of an operating session of the operating system to persist from one client station to another. The volatility or persistence of the storage space used for the storage of the written data is parameterizable when this would make sense. The volatile character of the redirections of the written data is determined upon initialization of the operating session of the operating system of a client station; if it is modified on the spur of the moment it will only affect the client station after a boot or reboot of the latter.

The data read requests issued by the operating system may be made in different storage spaces durng an operating session of the operating system. In this case, the data read requests issued by the operating system to an emulated hard disk made in different storage spaces are following an order of priority stemming from the previous order of the different redirections of the data write requests.

Said orders of priority depend on the sequence previously followed for the redirections of data write requests. In this way it is assured, from the standpoint of the client station, that the emulated hard disk is always coherent; in particular, if a particular datum D1 (a sector of an emulated hard disk or a file from a file system) was written in a data space El at a given moment, after which the data write requests were redirected to another space E2, then the same datum Dl (the

same sector or the same file) was modified and subsequently written in the space E2, the read requests on Dl then go to the space E2 (which has the highest priority), so as to read the most recently written version of Dl.

For example :

1. Are the data to be read found in the random access or virtual memory of the client station redirected to this storage space because of previous data write requests? If yes read them, if no

2. Are the data to be read in a buffer storage space (a file, for example) accessible to the peripheral drivers, which are part of the invention, of the client station due to previous data write requests redirected to this storage space? If yes read them, if no,

3. Are the data of the emulated hard disk stored on a server in a data processing network? If yes, go to item 4, if not, go to item 5.

4. Are the data to be read in a buffer storage space (a file, for example) accessible from a server module due to previous data write requests redirected to this storage space? If yes read them, if no,

5. Read the data directly in the support containing the unmodified data of the emulated hard disk.

Advantageously, if the hard disk emulation system is parameterized such that the data write requests received by the server software intended for a particular emulated hard disk are not redirected but stored directly in the support containing the data of the emulated hard disk itself, then a single client station may access said emulated hard disk at a given moment.

Therefore, the problems of concurrent access "in writing" to the same data set are avoided.

The parameterization for which the data write requests are not redirected is, e.g., the parameterizaton to be used for making available to client stations a new software component (a new software application, for instance):

- The parameterization of access to the emulated hard disk is positioned in such way that the data write requests are not redirected.
- The new software component is installed as if it were being installed on a real hard disk,
- The client station from which the installation was carried out is turned off,
- The parameter settings of access to the emulated hard disk are modified in such way that the data write requests of the client stations are redirected,
- All of the client stations that use the emulated hard disk in question from now on have access to the new software component without the necessity of making it explicitly available to each client station.

In order to allow several client stations to have simultaneous access to the same emulated hard disk the server software is able to redirect specifically the data write requests issued by client station A into a given storage space, e.g., a file in the storage space accessible to the server station, and to redirect the data write requests sent by another client station B into another given storage space, e.g., another file in the storage space accessible to the server station.

To accelerate the simultaneous access by several client stations to the same emulated hard disk, whose data are contained in a data support accessible to the server station, the data sent by the server station to the client stations within the scope of the hard disk emulation may be sent globally and at one time by using the "broadcast" or "multicast" mechanisms instead of the "unicast" mechanism.

The data sent by "broadcast" or by "multicast" by the server station are stored by the client stations that accept them in a local cache situated in the memory (real or virtual) of said client stations.

Advantageously, the client stations may delete the data cache that has been there more than a specified time and parameterizable (timeout). In fact, these data, if they are not pertinent and if they are never read, may uselessly plug up the "multicast" cache of the client stations.

A request for reading data in the emulated hard disk issued by the operating system of a client station generates an explicit data reading request issued to the server station only if said data are not already present in said local cache. If said data are effectively already present in said local cache they are read and transmitted to the operating system of the client station.

The read data in the local cache are withdrawn from it, after which they are read by the client station so as to free up the place in said local cache.

The decision to send, within the scope of the hard disk emulation according to the invention, of data by "multicast/broadcast" or "unicast" is made at the server module level which provides the functionalities necessary for the hard disk emulaton at the client stations.

The client stations may modify their subscription for receiving the data sent via "broadcast/multicast" by the server stations within the scope of the hard disk emulation according to the invention. For example, the client stations, once they have been started up from an emulated hard disk and [once] the man/machine interface permitting one or more users to make use of the client station is usable, the subscription for receiving data via "broadcast/multicast" can be cancelled, for example, by an unsubscription mechanism to a group of "multicast" addresses, so that the completely started up client stations no longer have to deal with the flood of "multicast/broadcast" data; this could prove to be very useful if the data sent by the server station via "multicast/broadcast" are only intended to permit access to portions of the emulated hard disk necessary for a complete startup of the operating system of the client station and are no longer useful to the client stations already started. Therefore, a client station already started up from an emulated hard disk is not polluted by the data sent by "broadcast/multicast" to the client stations during startup.

According to the invention, in order to permit the startup from and/or simultaneous access to the same emulated hard disk or to 100% identical copies of the same emulated hard disk, certain components constituting the invention

loaded and executed by the client stations, e.g., one or more peripheral drivers, or by a software server, are capable of modifying, on the fly or before their effective use by the operating system, certain data contained in the emulated hard disk, e.g., identifying only one client station on the network (computer name or IP address, for example) or the account password of the client station in an authentication field ("active directory" field, for example), serial numbers of software programs, or even the authentication data such as the data necessary to activate certain Microsoft products.

The emulation itself may be accomplished for the operating system of the client station at the level of the class of virtual peripherals of the file system type such as the products "Qualystem LiteNET PC 1.x and Qualystem LAN PC 2.x" (CIFS or SMB file System) or "Qualystem Rescue 1.x" (ISO9660/Joliet, CDFS or UDF file System).

The emulation may also be accomplished for the operating system of the client station at the level of the class of virtual disk peripherals itself and not at the level of the file system. This type of emulation is employed, for example, in the products "Qualystem LAN PC 3.x" (support for data of emulated hard disks residing on a server of a data processing network) or "Qualystem Rescue 2.x and 3.x" (support for data of emulated hard disks residing in the startup part called "El Torito" of an optical disk, CD or DVD).

The significant data contained in the emulated hard disk are copied by a software tool at a reference station from a real hard disk called the reference hard disk, accessible to the operating system of said reference station.

The software tool creates a file or an image directory (image file for emulation on a new hard disk, image directory for emulation on the file system level) which contains the data of the emulated hard disk. Said image file or image directory is created by making a copy of each file contained in the real hard disk used as a reference for creating the image file or image directory.

In the case of the image file, the latter is a bit for bit representation of a real hard disk considered as an array of bits. It is the logical representation of a hard disk in the form of the array of bits that is used. The image file thus contains the information necessary for the representation of the logic design of a real hard disk such as the primary BOOT sector (MBR, master boot record) and the list of partitions. It also contains the startup sector or sectors of partitions contained in the emulated hard disk. The software that creates the image file from a real hard disk can be dragged and dropped if the emulated hard disk is to be started, to modify in the data stored in the image file certain information used by the operating system loaded from the emulated hard disk in order to make the startup of said operating system from said emulated hard disk operational and effective.

In order to permit the startup from an emulated hard disk, the sequence of loading of the components of the operating system may require an adaptation so that all components of the operating system on which the peripheral drivers depend permit access to the emulated hard disk according to the invention are loaded and usable at the instant when the operating system needs to access the emulated hard disk by using the peripheral drivers and no longer using the "firmware" functions (BIOS).

For instance, if the data contained in the emulated hard disk used for startup of the operating system of a microcomputer are stored in a file on a server of a data processing network, the pilots permitting the operating system of the client station to use the network interface of said client station must be loaded and usable before the operating system needs to access the emulated hard disk via the peripheral drivers so as to continue loading; likewise, at this moment the network protocol pilots used by the peripheral drivers permitting access to the emulated hard disk must be loaded and usable.

Another example is that when the data contained in the emulated hard disk used for starting the operating system of a micro-computer are stored in the startup part of an optical disk (CD or DVD). In such a case, the peripheral drivers

permitting the operating system to access said optical disk reader must be loaded and usable before the operating system needs to access the emulated hard disk, via the peripheral drivers, in order to continue its loading.

It may therefore occur that the order of loading of the components on which the peripheral drivers permitting access to the emulated hard disk depend will have to be modified.

In the case when the data of an emulated hard disk are accessed by a client via a data processing network, the server module(s) potentially involved may make the emulated hard disks available on the network by utilizing a proprietary protocol such as that presently used in Qualystem LAN PC 3 or by using a suitable third party protocol, e.g., a "standard" protocol in order to export remote hard disks or file systems, for example, the protocol iSCSI (remote hard disk) or the SMB/CIFS or NFS protocols (remote file systems). The optional server module is then a service to which the clients who "speak" the protocol used may be connected and therefore access the data resources made available by said service. The possibility exists on the level of said service of managing the written data in parameterizable form and of being able, in particular, to provide in a storage space, specific to the client station/virtual hard disk pair, storage space that is not the common storage space of all clients. The clients that may be connected to the emulated hard disks made available by the server module on the network may therefore be "standard" clients, e.g., iSCSI standard clients or SMB/CIFS standard clients and not necessarily Qualystem^{TM} LAN-PC 3 clients, for example. Likewise, the programs executed by the client stations, low level software module, or peripheral drivers may use a proprietary network protocol to access the data contained in the emulated hard disks, such as the NVD protocol of Qualystem LAN PC 3 or a third party protocol, e.g., standard such as iSCSI or SMB/CIFS

Depending on whether the implementation at the level of the server modules or clients implements the functions of the present invention, the